Chapter 4

Methods of Diagnosis

Troubleshooting efforts are divided into two types: reactive and proactive. Reactive troubleshooting is efforts to diagnose and resolve reports of e-resource access disruptions received by libraries. These efforts are not planned, but in response to a previously unknown need. Conversely, proactive troubleshooting represents efforts that attempt to mitigate the daily influx of problem reports by identifying and resolving issues ahead of time. Unfortunately, many libraries are unable to dedicate a significant number of staff to engage in reactive troubleshooting. Proactive efforts, too, are often neglected due to a scarcity of staffing resources. As Rathmel and colleagues (2015) conclude from the results of their troubleshooting survey: out of 234 library respondents, 67 percent reported primarily troubleshooting with reactive approaches, and only 27 percent reported proactively troubleshooting. Consequently, libraries are looking for tools to diagnose and resolve access issues as efficiently as possible.

One of the significant gaps in library troubleshooting literature is methods of diagnosis. This is understandable given that each library system and electronic collection is unique. Both Rathmel and colleagues (2015) and Heaton (2018) ask the question "What are the best tools for troubleshooting?" Much of their focus is on tools that gather information, communicate resource status, and coordinate tasks. These include e-mail programs, ticket trackers and customer relationship management (CRM) systems, intranet applications, ERMSs, and project management platforms. While these tools are important for troubleshooting, forming the backbone of data collection and communication, they do not represent the best tools for diagnosing, which many novice troubleshooters seek. In this chapter, we attempt to fill this gap by reviewing trends and common methods of diagnosis for both reactive and proactive troubleshooting discussed in the literature and presented at conferences or workshops.

Troubleshooting Methodology: A Modified Version of Ross and Orr's DECSAR

Several troubleshooting methodologies already exist within the discipline of information technology, including the six-step methodology recommended by the Computing Technology Industry Association (CompTIA) and the DECSAR method, which was first developed by Ross and Orr (2009) to assist in the education and training of novice troubleshooters. We recommend a modified version of Ross and Orr's DECSAR method in our book *The Electronic Resources Troubleshooting Guide* (Talbott and Zmau 2020). It specifically addresses the library troubleshooting needs around communication, assessment, and documentation and consists of these seven stages:

1. Identify and define the problem.
2. Examine the situation.
3. Consider the possible causes.
4. Consider the possible solutions.
5. Implement the solution.
6. Review the results.
7. Communicate and document the resolution.

When compared to other troubleshooting methodologies, the DECSAR method in particular is notable for its depiction of the iterative nature of troubleshooting. This method identifies both the ideal linear path, which is often the sole focus of other troubleshooting methodologies, and the backtracking, or recursive thinking, that is necessary depending on the complexity of the issue and the skill of a troubleshooter. As we

discuss reactive and proactive troubleshooting efforts throughout this chapter, we will concern ourselves with step number 3 of our DECSAR troubleshooting method and consider the possible causes.

Reactive Troubleshooting

Techniques for Diagnosis

Troubleshooting diagnosis can be overwhelming. As Emery, Stone, and McCracken (2020) explain, when attempting to diagnose an access disruption, "it is impractical to check all of the following variables, and especially to check all variables against all other variables" (97). When confronted with all the possible variables to check, novice troubleshooters often do not know where or how to begin. Along with the DECSAR method, the following techniques can be used by troubleshooters to jump-start their diagnosis:

- *Re-creation:* The troubleshooter isolates the cause of the issue by finding a procedure (sequence of steps or events) that consistently induces the symptoms or failure to occur.
- *Elimination:* The troubleshooter isolates the cause of the issue by systematically testing and eliminating possible causes.
- *Backtracking:* The troubleshooter isolates the cause of the issue by starting at the point of system failure and reasoning backward, testing each possible cause along the way (Gugerty 2007).
- *Half-splitting:* Using this method, the troubleshooter divides the system into portions and checks each portion for symptoms of the issue; this procedure is repeated in the portion where the symptoms occur (by again dividing and testing each half) until the cause of the issue has been isolated.

Re-creation is perhaps the most widely employed and talked about method for e-resource troubleshooting. In their book *The ABCs of ERM*, Zellers, Adams, and Hill list it as the first step in their four-part troubleshooting process, and also one that frontline staff should be trained to do before transferring problems to the specialists (Zellers, Adams, and Hill 2018, 158). Other articles, including Davis and colleagues (2012), Browning (2015), Hart and Sugarman (2016), and Shriver (2019), also mention this method, usually couched within the context of tools meant to facilitate the re-creation process. By attempting to replicate an issue, the troubleshooter gains essential contextual information that verbal or written descriptions simply cannot provide. Often, this information is enough to pinpoint the cause of the access issue without the need for additional testing or investigation. Therefore, the very first troubleshooting strategy we recommend that all troubleshooters try when diagnosing an access issue is re-creation.

For instance, whether the source of an access issue is within a patron-controlled component (e.g., device, browser, or internet connection) or within a library- or vendor-controlled component (e.g., catalog, knowledge base, discovery service) will affect whether the troubleshooter is able to reproduce the issue.

As a general rule of thumb, reproducible problems indicate that the cause of the breakdown is within the library-controlled or vendor-controlled part of the access chain. This makes sense, of course. If an issue is presenting itself to multiple users (in this case, the reporter and the troubleshooter) who are employing different devices, browsers, and network settings, the issue is likely unrelated to these patron-specific components. There are exceptions to this rule, of course, mostly in regard to browsers and browser settings, which can be reproduced if the troubleshooter is given enough information.

Elimination is another method frequently discussed in the literature, although usually without naming it as such. The suggested strategies usually entail testing certain components, such as browsers, caches and cookies, or devices, by replacing or removing them from the access chain to see if the issue reappears. This could mean asking patrons to use a different device or browser, to clear their browser's cache and cookies, to disable any advertising or pop-up blockers, or to modify their security settings. This also means being aware of any compatibility issues or software requirements for specific vendor platforms. If a vendor platform is not compatible with mobile devices or can be accessed using only certain browser versions or DRM software, the troubleshooter should first check that the patron is meeting these requirements before diving into additional problem solving. For instance, Emory, Stone, and McCracken developed a browser rubric to systematically test popular browsers on Macs and PCs from both on and off site (Emory, Stone, and McCracken 2020, 103). By diligently testing each combination, a troubleshooter can gain a comprehensive view of which factors—browser, device, location, or a combination—contribute to the appearance of the access failure.

Both re-creation and elimination are extremely useful in isolation; however, they alone cannot solve every issue. Sometimes, they will need to be used in conjunction with other troubleshooting methods to pinpoint the cause of an access disruption. Consider this example of backtracking: a troubleshooter receives a problem report from a professor who cannot access an e-resource from within the learning management system (LMS). Through the troubleshooting interview, the troubleshooter learns that the professor had embedded one of the library's research guides within a course page, but when the professor

clicked on the link to a database included in the guide, he received a 404 error message. Presuming the troubleshooter has access to the course page, the troubleshooter would first attempt to replicate the issue by navigating to the embedded guide and clicking the link. (Alternatively, the troubleshooter could navigate directly to the research guide from the library website to test the link there.) The troubleshooter receives the same 404 error message. After taking a closer look at the link, the troubleshooter discovers it is a friendly URL originating from the library's database A–Z list. The troubleshooter then navigates to the database A–Z list and tests that link. It, too, produces a 404 error message. Logging into the back end of the A–Z list, the troubleshooter compares the entry's URL (the one masked by the friendly URL) to that currently being used on the database's home page. The URLs are different. The troubleshooter updates the database entry to use the current home page URL, which resolves the issue and allows the professor to access the database from within the LMS.

While re-creating the issue was an essential first step in diagnosis, replication alone was insufficient to pinpoint the cause. Because the URL of the link was passed through three access tools (database A–Z list to research guide to LMS course page) as well as hidden behind a friendly URL, simply finding and testing the link revealed very little as to why it was broken. Instead, the troubleshooter needed to follow the data back to its original source, testing along the way. This process of moving from LMS to research guide to database A–Z list is a great example of backtracking, which is most useful when trying to trace the origins of faulty metadata.

Half-splitting (also called chunking or the divide-and-conquer method) is another useful strategy for isolating the cause of an access issue. Using this method, the troubleshooter divides a system into segments (traditionally, into halves, thus the term *half-splitting*) and tests each segment for signs of the problem, repeating the process of dividing and testing until the faulty component is identified. Half-splitting is most effective when the troubleshooter is uncertain which area of the access chain an issue is originating from and wants to systematically test each portion, rather than randomly testing or eliminating components.

For instance, let's imagine a troubleshooter is trying to determine why an off-campus patron is receiving a timeout error message when accessing an e-resource via the library's e-journal A–Z list. The troubleshooter could mentally divide the access chain into two halves: the patron-controlled components and authentication in one half and the e-journal A–Z list and vendor platform in another. To test the first half, the troubleshooter could provide the patron with a proxied link to a test e-resource—one the troubleshooter knows works correctly—and ask the patron to attempt to access the content. To test the second half, the troubleshooter navigates to the faulty e-journal within the A–Z list and attempts to access it firsthand. The troubleshooter is successfully able to view the journal on the vendor platform, but the patron reports that using the test link caused the timeout error message to appear.

Since the issue appeared again for the patron but not the troubleshooter, the next step would be to divide the first segment into its distinct pieces to test authentication and the patron-controlled components separately. To test the authentication, the troubleshooter could ask the patron to attempt to access an e-resource using the WAYF menu on a federated SSO-enabled vendor platform. This bypasses the proxy server entirely, while allowing the patron to use the same device, browser, and network connection as before. To test the device, browser, and network connection, the troubleshooter could first ask whether the patron has experienced connection issues, such as slow load times, while using nonlibrary web resources. The troubleshooter may also have the patron assess personal connection speed using a free online tool. However, the troubleshooter would likely want to wait for the results of the WAYF test before diving too far into such tests.

The patron reports being able to access the content using the WAYF menu and having no connection issues while browsing the web. This implies that some negative interaction with the proxy server (i.e., authentication) is at fault. Based on experience, the troubleshooter knows that browser cookies often interact negatively with the proxy server, so the troubleshooter would likely ask the patron to clear the cache and cookies as well as browsing history or attempt to use another browser entirely to see if either resolves the issue. If the issue persists or if more patrons report similar errors, the troubleshooter may want to consult with the staff who manage the proxy server in order to come to a satisfactory resolution, such as modifying the configurations or updating the server software.

Again, in this example, re-creation by itself was of limited value to the troubleshooter. Since the issue appeared for the patron but not the troubleshooter, it was difficult for the troubleshooter to gauge which components were contributing to the issue and which were extraneous. Similarly, while elimination may have eventually isolated the cause of the error, testing individual components in a sequential order—or, worse, jumping between components randomly or whenever inspiration strikes—is inefficient and can lead to frustration as time drags on. By chunking the access chain into segments and systematically testing each one, the troubleshooter was able to quickly home in on the culprit.

Tools for Diagnosis

GENERIC TOOLS

What's My IP

Knowing the exact IP address for an end user's computer can be helpful to quickly determine if they are located on or off site. Similarly, troubleshooters often provide their IP address to vendors to help troubleshoot systemic issues. An originating IP address can be determined by querying Google "What's my IP?" or by visiting websites such as WhatIsMyIPAddress.

WhatIsMyIPAddress
https://whatismyipaddress.com

Incognito Mode

Troubleshooters can use incognito mode to test whether a browser's cache, cookies, or browsing history is interfering with access. This feature is referred to as Incognito Mode in Chrome, Private Browsing in Firefox, InPrivate Browsing in Microsoft Edge, and Private Window in Safari. Incognito mode works by creating a separate, "clean" session within a browser, free from any previously stored web data. This allows troubleshooters to effectively clear their caches and cookies without actually losing any of the information.

Verifying If a Website Is Down

There are several websites that can verify if a web page is down for just your PC (personal computer) or for everyone. These websites allow you to plug in a URL to see how many users are affected by a vendor's potential downtime.

Downforeveryoneorjustme.com
https://www.downforeveryoneorjustme.com

IsItDownRightNow?
https://www.isitdownrightnow.com

Solutions Repository

Some libraries have found success by implementing local solutions repositories to address regular troubleshooting issues. A solutions repository can take the form of a blog, wiki, Word document, research guide, and so on. Its purpose is to proactively list solutions to common, simple, or known issues so that troubleshooters can respond more quickly to access issues. Some commercial ticketing systems have features similar to a solutions repository. For example, Springshare's LibAnswers provides an option to reuse previous ticket answers called Reuse Answers. Examples for what to incorporate into a solutions repository include e-resources that often confuse users, tricky technology configurations, e-resources that require individual accounts for access, and e-resources or vendors known for lengthy downtimes.

A solutions repository can be useful both for troubleshooting staff and for other staff whose primary duties may not include troubleshooting. As Samples and Healy highlight, "Public-facing staff can do access problem triage by consulting the wiki and getting back to the patron with an explanation without having to submit a form or an email to the troubleshooting team" (Samples and Healy 2014, 114).

Screen Capturing Programs

Screen capturing programs like Snipping Tool (Windows), TechSmith Capture, or Snagit (TechSmith) are frequently mentioned throughout the literature and can be invaluable for troubleshooters who struggle to see exactly what their reporters are reporting. By requesting a screenshot from a user, a troubleshooter can identify pertinent details from the image, such as if the user's browser URL field includes proxy details or if the user is appropriately signed in to an e-resource with their account. Troubleshooters also use screenshots to communicate more easily with vendors and to capture detailed instructions for users that would otherwise be cumbersome to communicate via e-mail.

TOOLS FOR TROUBLESHOOTING OPENURL LINK RESOLVER ISSUES

Carter and Traill facilitated a workshop called Teaching Troubleshooting at the 2019 Electronic Resources and Libraries Conference in Austin, Texas. The workbook they provided for the workshop includes a list of practical diagnosing tools such as extensions for HTTP headers, options for parsing OpenURLs, and local bookmarklets for revealing discovery service source records (Carter and Traill 2019). These tools are highly relevant and practical for troubleshooting OpenURL issues.

Parsing OpenURLs into their individual components can make them more easily readable, aiding in identifying faulty metadata for quicker resolution. Carter and Traill (2019) specifically mention Jeff Peterson's OpenURL deconstructor (Peterson, n.d.) and the FreeFormatter URL Parser/Query String Splitter (FreeFormatter, n.d.). These tools can be used to determine if incorrect metadata is causing an OpenURL link to fail. Relatedly, various browser extensions can be employed to view the HTTP headers in a browser. HTTP headers display the header name and value, often separated by a single colon, for both HTTP requests and responses. For troubleshooting, viewing HTTP headers can reveal the exact requests

and responses that are being processed in real time when attempting to access an e-resource. This in turn can aid in spotting if proxy details are being appropriately passed or where exactly a connection is being lost when a link fails.

Some libraries have developed their own tools that can be used to troubleshoot common issues within their local systems. Carter and Traill (2019) specifically mention that bookmarklets are beneficial for troubleshooting their local system Ex Libris's Alma LSP. A bookmarklet is a web browser bookmark that contains JavaScript commands that add new features to the browser. Ex Libris provides a Developer Network for its customers where information and technology solutions can be shared in benevolence among the Alma community. For example, the bookmarklet PrimoNUIShow is used to display the RecordID and PNX source record for a record from the Primo discovery search (Höfler 2018). The PNX record can be very useful when troubleshooting Primo discovery search issues because it reveals the source of the original record and therefore reveals who should be contacted, be it the vendor, Ex Libris, or the local library, when an issue is encountered.

TOOLS FOR TROUBLESHOOTING REMOTE ACCESS

The cornerstone for troubleshooting remote access is simulating off-site access while being on site. This is usually done by utilizing on-site technology with an IP address not included in the institution's on-site ranges. There are several different technology options to achieve this goal. Rodriguez, Tonyan, and Wilson (2018) include a list of practical diagnosing tools for troubleshooting remote access, such as smartphones connected to a cellular network, mobile Wi-Fi hotspots, remote desktop solutions, commercial (not institutional) VPNs, and advanced troubleshooting options for EZproxy. We review a few of these tools here.

Smartphones connected to a cellular network are often a first stop for troubleshooters looking to troubleshoot remote access; however, there are downsides to this. Using a personal device for work purposes is an imperfect solution, and a strong cellular network connection is required. Depending on the volume of troubleshooting needed and the library staff member's data plan, data costs may be significant. Moreover, access disruptions may pertain to specific devices or operating systems that are unable to be tested on a smartphone.

Personal devices can also be used for troubleshooting when connected to mobile Wi-Fi hotspots. Mobile hotspots are considered affordable both for initial purchase cost and annual data rates, and they can be especially handy for troubleshooters whose libraries already offer hotspot devices for checkout to their users. However, some institutions may restrict work machines from logging on to external wireless networks.

Beyond a smartphone or mobile Wi-Fi hotspot, remote desktop solutions are another common option. Remote desktop solutions, such as Microsoft Remote Desktop, TeamViewer, and Chrome Remote Desktop, allow a desktop to be controlled from another computer or device (Rodriguez, Tonyan, and Wilson 2018). External VPNs are also a very popular troubleshooting solution; however, some commercial VPNs are subject to increased scrutiny from institutional IT departments, which may or may not restrict staff from being able to download and install the commercial VPN client on the internal network.

Several resources exist for troubleshooting remote user access issues originating from the institution's proxy server. They include access to the local configuration files, OCLC's EZproxy Database Stanzas list, the EZproxy LISTSERV, and configuration file directives that can be used by EZproxy administrators. Many proxy issues can be resolved by updates to the e-resource's stanza within the configuration file. With access to the local configuration file, troubleshooters can swiftly enact these updates. OCLC manages a list of current stanzas on its EZproxy Database Stanzas list. Troubleshooters can use this list to verify what the most current stanza for an e-resource platform should be. The OCLC EZproxy Database Stanzas list should be the first place troubleshooters check after verifying a proxy prefix has been applied appropriately to a resource URL. E-resource platform vendors are also able to provide stanzas for their products upon request. Additionally, the EZproxy LISTSERV is an invaluable resource for being proactively notified of proxy issues with major vendors. The LISTSERV also provides an outlet for discussion and the opportunity to learn how to better resolve issues from other, more seasoned professionals. Finally, the EZproxy RequireAuthenticate and MinProxy directives can be used for simulating remote access. More information about these directives can be found in OCLC's online support portal.

Proactive Troubleshooting

One of the avenues for attempting to reduce the number of problem reports received is to analyze problem reports to identify systemic issues. Another is to conduct routine access checks for a library's holdings. Both of these proactive methods can be helpful in reducing both user and librarian frustration. Analyzing problem reports can reveal a library's most common issues, uncover previously unknown underlying issues, and help a troubleshooting team determine where they can correct their course to better invest their limited resources. Conducting routine access

checks can discover broken links before users are able to report them. Proactive troubleshooting also encompasses working with access tool or e-resource vendors to address known issues via the vendor website, e-mail, phone, or in-person communication, as well as working with peer librarians whose libraries use similar products.

Analyzing Problem Reports

Analyzing problem reports can directly tie in to workflow assessment because so many different variables can be evaluated against initial assumptions or preconceived notions. Many libraries choose to analyze their problem reports with these goals in mind (Lowry 2021):

- to identify common points of failure
- to identify any underlying, systemic issues with particular e-resources, authentication methods, access tools, or user groups
- to determine if additional staffing resources are needed due to ticket volume
- to identify gaps in the troubleshooting workflow
- to inform collection development decisions
- for training purposes

There is much discussion in the literature concerning the need for a controlled vocabulary when analyzing problem reports. Individual libraries can analyze their own tickets using their own terms and categorization methods. However, in order for an individual library to gain context for the results of its local analysis, a shared language of description would be necessary. Gould and Brett (2020) note that currently further standardization and agreed-upon definitions would be required to see the true value in a shared vocabulary. Both Gould and Brett (2020) and Goldfinger and Hemhauser (2016) propose that a NISO standard be created for categorizing e-resource access issues. As Gould and Brett state, "Developing a NISO standard would be the best way to codify Goldfinger and Hemhauser's problem types and functional areas and lead to wide adoption within the library community" (Gould and Brett 2020, 198). Some libraries are adopting the controlled vocabulary, called "Functional Areas," developed by Goldfinger and Hemhauser (2016). Examples of Goldfinger and Hemhauser's Functional Areas include KB/link resolver, proxy/IP problems, incorrect URL, excessive usage/downloading, and subscription problem.

Brett (2018) used Goldfinger and Hemhauser's (2016) controlled vocabulary, and Lowry (2020) then built upon the work of Goldfinger and Hemhauser (2016) and Brett (2018) in her own ticket analysis in order to identify trends that could signal an industry-wide issue. The study confirmed that "certain types of access problems do occur at similar rates among research institutions, despite the likely differences in workflows, tools, management styles, and varying collections among them" with the "two most common problems at all three libraries f[alling] into the categories of KB/Link Resolver or Platform" (Lowry 2020, 29). To supplement this analysis, Lowry also created a locally defined schema to illuminate access disruption trends affecting her institution. Example categories include concurrent user limits, bad record in catalog, DDA problem, referral to another department, and accessing canceled titles/resources.

Access Checks

As libraries subscribe to or purchase e-resources, a record is usually created to serve as a receipt for what the library is gaining access to and for how long. These receipts usually take the form of title lists from licenses, subscription agent interfaces, or vendor administration portals. Acquisition records can also provide more details. Title lists from these sources usually include a title, a unique identifier (such as ISBN or ISSN), publisher or vendor information, coverage dates, and the website where the licensed content can be accessed. Subscription coverage dates vary widely from vendor to vendor, and libraries have not always licensed perpetual access to content they pay for. To determine whether the library retains perpetual access or post-cancellation access, a troubleshooter would need to consult the resource's specific license terms.

Although librarians do their best when making e-resources available for discovery, inaccuracies inevitably occur in a library's holdings long after the e-resources were originally acquired. Especially for e-journals, which can experience changes in title, publisher, and hosting provider, the very nature of subscriptions introduces variables that can cause a library's access to be inadvertently cut off. A vendor, for example, may accidentally disable a library's access even after a renewal invoice is paid or may fail to reestablish access after receiving a late payment. One-time purchases can also experience similar acquisitions issues due to continuing access fees.

In response to these issues, some libraries perform systematic access checks with the goal of comparing a reliable access list, such as a vendor title list, to what has already been enabled within the library's discovery system. An access check for a single subscription ensures that the correct title, coverage dates, and platform for access are available for users to discover. A vendor title list can be cross-referenced with acquisitions data when available, and acquisitions data alone can be used if a vendor does not provide a title list of subscriptions or one-time purchases.

Mortimore and Minihan (2018) go into great detail about how often they conduct access checks for

various types of e-resources and why. Their troubleshooting staff perform bi-weekly database link audits and quarterly link asset audits and have a rolling link resolver audit. When prioritizing where to start when beginning e-resource access checks, many libraries consider the following:

- Any known, systemic issues: For example, a library has received multiple problem reports that it is missing access to titles on a single e-journal platform. Until the troubleshooter obtains a title list for the vendor platform, the troubleshooter will be unable to correct these issues en masse.
- A library's major vendors: For example, if the vast majority of a library's holdings are held between five to six vendors, it would be best to start with them.
- A library's most popular resources: Prioritize by subject discipline, audience size, or highest usage.
- Any obvious discrepancies in what a library should have access to and what a library cataloged: If a title list retrieved from a publisher states that a library should have access to twenty titles but instead the library has fifty titles cataloged, this should likely be examined sooner rather than later.

Spot-checking is also an alternative if a library does not have the staff time to devote to checking title lists, and so on, in their entirety. With limited staff time, a troubleshooter could check 20 percent of a library's holdings to determine if checking the remaining 80 percent is warranted. Not all title lists need to be checked individually in their entirety. It can also be argued that single title subscriptions that are dynamic by nature, such as e-journals and e-book packages, should probably be checked more frequently than one-time purchases that are static by nature, such as databases, e-books, and streaming videos.

Another remedy for limited staff time is to take advantage of any available link-checking features offered by access tools. For example, Springshare offers an automated link-checking tool that libraries can use to find broken links in both LibGuides and LibGuides A–Z Database List. There are also other link-checking tools that you may be able to use at your library, such as Callisto (Headlee and Lahtinen 2014; Sharp Moon 2017) or SEESAU (Serials Experimental Electronic Subscription Access Utilities; Collins and Murray 2009). In addition, it is common for libraries to use homegrown link checkers, often utilizing Python programming language, that are developed either by IT departments or by troubleshooting librarians who are familiar with coding.

Even without link-checking tools, access checks can be conducted periodically by a staff member. Access checks are usually simple enough that they can be assigned to student workers or to any other staff members who are unfamiliar with e-resources. Generally, once shown the basic requirements of link-checking, these staff members will succeed.

References

Brett, Kelsey. 2018. "A Comparative Analysis of Electronic Resources Access Problems at Two University Libraries." *Journal of Electronic Resources Librarianship* 30, no. 4: 198–204. https://doi.org/10.1080/1941126X.2018.1521089.

Browning, Sommer. 2015. "Data, Data, Everywhere, nor Any Time to Think: DIY Analysis of E-resource Access Problems." *Journal of Electronic Resources Librarianship* 27, no. 1: 26–34. https://doi.org/10.1080/1941126X.2015.999521.

Carter, Sunshine, and Stacie Traill. 2019. "Teach Your Staff to Troubleshoot E-resources: Practical Processes for Documenting and Implementing a Troubleshooting Training Curriculum." Workshop presented at the 2019 Electronic Resources & Libraries Conference, Austin, TX, March 6, 2019.

Collins, Maria, and William T. Murray. 2009. "SEESAU: University of Georgia's Electronic Journal Verification System." *Serials Review* 35, no. 2: 80–87. https://doi.org/10.1080/00987913.2009.10765216.

Davis, Susan, Teresa Malinowski, Eve Davis, Dustin MacIver, Tina Currado, and Lisa Spagnolo. 2012. "Who Ya Gonna Call? Troubleshooting Strategies for E-resources Access Problems." *Serials Librarian* 62, no. 1–4: 24–32. https://doi.org/10.1080/0361526X.2012.652459.

Emery, Jill, Graham Stone, and Peter McCracken. 2020. *Techniques for Electronic Resource Management: TERMS and the Transition to Open*. Chicago: ALA Editions. https://pdxscholar.library.pdx.edu/cgi/viewcontent.cgi?article=1305&context=ulib_fac.

FreeFormatter. n.d. "URL Parser/Query String Splitter." https://www.freeformatter.com/url-parser-query-string-splitter.html.

Gugerty, Leo. 2007. "Cognitive Components of Troubleshooting Strategies." *Thinking and Reasoning* 13, no. 2: 134–63. https://psycnet.apa.org/doi/10.1080/13546780600750641.

Goldfinger, Rebecca Kemp, and Mark Hemhauser. 2016. "Looking for Trouble (Tickets): A Content Analysis of University of Maryland, College Park E-resource Access Problem Reports." *Serials Review* 42, no. 2: 84–97. https://doi.org/10.1080/00987913.2016.1179706.

Gould, Elyssa M., and Kelsey Brett. 2020. "A Tale of Two Universities: Electronic Resources Troubleshooting Comparisons." *Serials Librarian* 79, no. 1–2: 192–99. https://doi.org/10.1080/0361526X.2020.1760184.

Hart, Katherine A., and Tammy S. Sugarman. 2016. "Developing an Interdepartmental Training Program for E-resources Troubleshooting." *Serials Librarian* 71, no. 1: 25–38. https://doi.org/10.1080/0361526X.2016.1169569.

Headlee, Patricia A., and Sandra C. Lahtinen. 2014. "Callisto Basic and Calisto Pro." *Journal of the Medical Library Association* 102, no. 4 (October): 305–6. https://doi.org/10.3163%2F1536-5050.102.4.018.

Heaton, Robert. 2018. "Tools for Troubleshooting: Which Ones and What For." *Journal of Electronic Resources Librarianship* 30, no. 1: 9–26. https://doi.org/10.1080/1941126X.2018.1443903.

Höfler, Gabriele. 2018. "New UI Bookmarklet: Show RecordID, PNX, Source Record." *Tech Blog*, March 22, 2017; updated December 18, 2018. Ex Libris Group. https://developers.exlibrisgroup.com/blog/New-UI-Bookmarklet-Show-RecordID-PNX-Source-Record/.

Lowry, Lindsey. 2020. "Where Do Our Problems Lie? Comparing Rates of E-access Problems across Three Research Institutions." *Serials Review* 46, no. 1: 26–36. https://doi.org/10.1080/00987913.2020.1733173.

———. 2021. "Fighting an Uphill Battle: Troubleshooting Assessment Practices in Academic Libraries." *Library Resources and Technical Services* 65, no. 1: 4–13. https://doi.org/10.5860/lrts.65n1.4-13.

Mortimore, Jeffrey M., and Jessica M. Minihan. 2018. "Essential Audits for Proactive Electronic Resources Troubleshooting and Support." *Library Hi Tech News* 35, no. 1: 6–10. https://doi.org/10.1108/LHTN-11-2017-0085.

Peterson, Jeff. n.d. "OpenURL Parser." https://gpeterso.github.io/url-params/.

Rathmel, Angela, Liisa Mobley, Buddy Pennington, and Adam Chandler. 2015. "Tools, Techniques, and Training: Results of an E-resources Troubleshooting Survey." *Journal of Electronic Resources Librarianship* 27, no. 2: 88–107. https://doi.org/10.1080/1941126X.2015.1029398.

Rodriguez, Michael, Joel Tonyan, and Robert T. Wilson. 2018. "Tools and Techniques for Troubleshooting Remote Access." *Journal of Electronic Resources Librarianship* 30, no. 3: 171–78. https://doi.org/10.1080/1941126X.2018.1494095.

Ross, Craig, and R. Robert Orr. 2009. "Teaching Structured Troubleshooting: Integrating a Standard Methodology into an Information Technology Program." *Educational Technology Research and Development* 57, no. 2: 251–65. https://doi.org/10.1007/s11423-007-9047-4.

Samples, Jacquie, and Ciara Healy. 2014. "Making It Look Easy: Maintaining the Magic of Access." *Serials Review* 40, no. 2: 105–17. https://doi.org/10.1080/00987913.2014.929483.

Sharp Moon. 2017. "Callisto." https://sharpmoon.com/callisto/.

Shriver, Emery. 2019. "Managing Discovery Problems with User Experience in Mind." *Code4Lib Journal*, no. 44 (May). https://journal.code4lib.org/articles/14481.

Talbott, Holly, and Ashley Zmau. 2020. *The Electronic Resources Troubleshooting Guide*. Chicago: ALA Editions.

Zellers, Jessica, Tina M. Adams, and Katherine Hill. 2018. "I Think the Internet Broke." In *The ABCs of ERM: Demystifying Resource Management for Public and Academic Librarians*, 154–168. Santa Barbara, CA: Libraries Unlimited.